22AIE114- INTRODUCTION TO ELECTRICAL AND ELECTRONICS ENGINEERING
22AIE115- USER INTERFACE DESIGN

ENERGY THEFT DETECTION AND POWER CONSUMPTION MONITORING OF ELECTRICAL APPLIANCES IN SMART GRID

GROUP- 5

- ANAKHA S- CB.SC.U4AIE24006
- CHAITHANYA G NAMBIAR-CB.SC.U4AIE24013
- SURABHI SAHA-CB.SC.U4AIE24057
- SRIRAM S - CB.SC.U4AIE24066

INTRODUCTION

Energy Theft and Inefficiency

Unauthorized electricity usage leads to financial losses and grid instability, making energy theft detection crucial.

Smart Grids and AI Integration

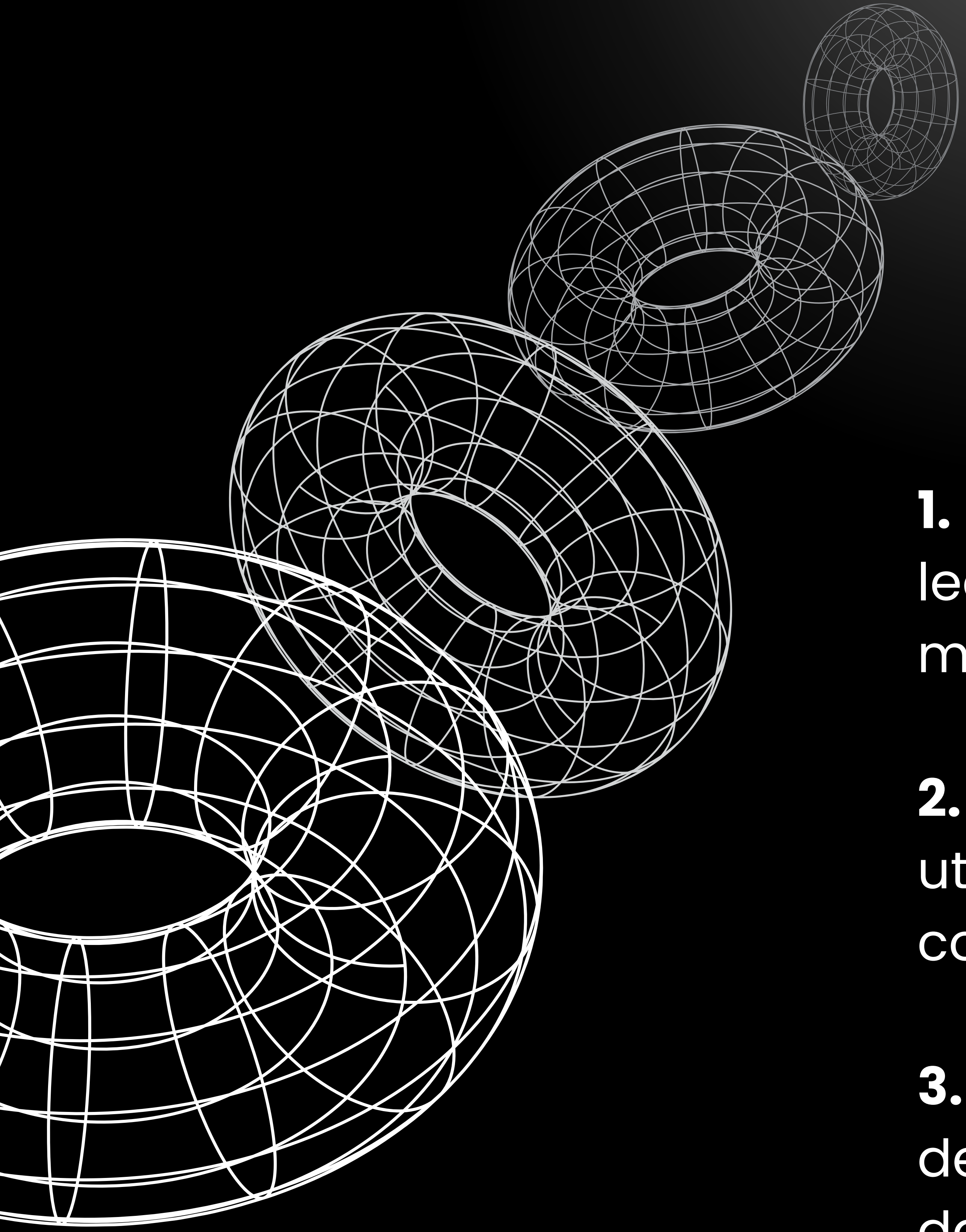
By leveraging deep learning models like Autoencoders and CNNs, smart grids can analyze power consumption and detect anomalies.

Real-Time Monitoring for Efficiency

A Flask-based dashboard and AI-driven analytics provide real-time insights into appliance-level power usage and theft detection.

OBJECTIVES

1. Detect Energy Theft – Identify unauthorized power usage using deep learning models.
2. Monitor Appliance-Level Consumption – Use NILM to track individual device power usage.
3. Enhance Smart Grid Security – Detect anomalies in energy patterns to prevent fraud.
4. Enable Real-Time Tracking – Develop a dashboard for live energy monitoring and alerts.
5. Ensure Data Accuracy – Preprocess smart meter data to remove noise and missing values.
6. Improve Grid Efficiency – Provide insights for better energy distribution and management
7. Optimize Energy Utilization – Reduce wastage by analyzing consumption patterns.



PROBLEM STATEMENT & SOLUTION

1. Problem: Energy Theft and Inefficiency – Unauthorized power usage leads to financial losses, grid instability, and inefficient energy management.

2. Problem: Lack of Appliance-Level Monitoring – Consumers and utility providers lack detailed insights into individual appliance consumption, making it hard to optimize usage.

3. Solution: AI-Powered Smart Grid System – By using smart meters, deep learning models (Autoencoders, CNNs), and a real-time dashboard, the system detects energy theft, monitors appliance-level consumption, and optimizes power distribution efficiently.

METHODOLOGY

Data Simulation & Collection: Smart meter data will be simulated for training and testing.
Data Preprocessing: Noise removal and missing value handling using Pandas & Scikit-learn.

Energy Theft Detection:

How It Works:
An autoencoder is a type of neural network that learns to compress data into a smaller representation and then reconstruct it back to the original form. It is mainly used for anomaly detection by identifying patterns that do not match normal behavior.
They learn normal energy usage patterns by compressing and reconstructing input data.
When unusual energy consumption (theft) occurs, the reconstruction error is high, indicating an anomaly.

Implementation Steps:
• Data Collection: Use Smart Meter data (simulated or real) containing voltage, current, power factor, and consumption.
• Preprocessing: Use Pandas & Scikit-learn to clean data, handle missing values, and normalize values.
• Building Autoencoder Model:
• Use TensorFlow/Keras to create an autoencoder with an encoder (compresses data) and a decoder (reconstruct data).
• Train the model on normal energy usage patterns.
• Anomaly Detection: Compare real-time energy data with expected patterns. If the reconstruction error exceeds a threshold, mark it as potential energy theft.
Alerts & Visualization: Display anomalies on a Flask dashboard and send alerts.

METHODOLOGY

Power Consumption Disaggregation Using NILM (Neural Networks):

How It Works:
NILM (Non-Intrusive Load Monitoring) → A technique or methodology used to break down total power usage into individual appliances without using separate sensors. It applies machine learning or deep learning models (like CNN, RNN, or Transformers) to analyze smart meter data and estimate each appliance's power consumption.
A Neural Network (CNN/LSTM) learns energy signatures of different appliances from smart meter data.
The model predicts which devices are running and their power consumption.

I**mplementation Steps:**
• Data Collection: Simulate or use real Smart Meter data with timestamps and total power consumption.
• Preprocessing: Use Pandas & NumPy to clean and normalize energy data
• Building Neural Network Model:
How CNN Works in NILM:

- Input: Aggregated power consumption recorded over time as a sequence (like an image).
- Feature Extraction: Convolutional layers detect appliance activation patterns, while pooling layers reduce noise.
- Output: The trained CNN estimates individual appliance consumption from the total power usage.

• Training the Model: Train with labeled appliance-level consumption data. Validate performance using accuracy metrics (MAE, RMSE).
• Prediction & Visualization: The trained model takes total energy input and predicts appliance-wise consumption. Display results in a Flask dashboard for real-time monitoring.

POWER CONSUMPTION OF APPLIANCES AND ENERGY THEFT DETECTION USING SMART GRIDS

1. Smart Grid Technology – A smart grid is an advanced electricity network that uses digital technology to monitor, control, and optimize power distribution efficiently.
2. Smart Meter Data Collection – Smart meters track voltage, current, and power factor in real-time, enabling precise monitoring of household energy consumption.
3. Appliance-Level Consumption Monitoring – Non-Intrusive Load Monitoring (NILM) with deep learning separates total power usage into individual appliances for better insights.
4. Energy Theft Detection with AI – Autoencoders analyze normal power consumption patterns and detect anomalies, such as sudden spikes, indicating potential energy theft.
5. Real-Time Analysis and Grid Optimization – The system continuously monitors energy usage, generates alerts for unusual patterns, and enhances overall grid efficiency by preventing theft and reducing wastage.

DATASET DETAILS

1.Simulated Smart Grid Data – The dataset includes power consumption data for 5 houses over 30 days, covering multiple electrical appliances.

2.Key Parameters Collected – The dataset records voltage, current, power factor, total power consumption, and energy theft, along with expected and actual usage for each appliance.

3.Appliance-Level Monitoring – Power usage data is generated for six appliances: Fridge, AC, TV, Lights, Washing Machine, and Microwave, allowing individual consumption analysis.

4.Energy Theft Simulation – Random variations in energy usage simulate potential energy theft, where extra power consumption is artificially introduced to test anomaly detection.

5.Data Format and Storage – The generated dataset is saved as a CSV file (smart_grid_power_usage.csv), making it easy to process and analyze using Python libraries like Pandas and TensorFlow.

FLOWCHART FOR ENERGY THEFT DETECTION AND POWER CONSUMPTION MONITORING

1. Start
2. Smart Meter Data Collection
3. Collect real-time voltage, current, power factor, and total power usage.
4. Data Transmission
Send collected data to a central processing unit via a two-way communication network.
5. Data Preprocessing
Remove noise, handle missing values, and normalize data for analysis.
6. Energy Theft Detection
Train Autoencoder model on normal usage patterns.
Decision:
If anomaly detected → Flag as potential energy theft → Send alert.
If no anomaly → Proceed to next step.

FLOWCHART FOR ENERGY THEFT DETECTION AND POWER CONSUMPTION MONITORING

7. Appliance-Level Power Monitoring
Use a CNN model to disaggregate total power into individual appliance consumption.
8. Dashboard & Alerts
Display power usage insights in a Flask-based UI.
Send notifications for abnormal energy consumption.
9. Decision & Action
Authorities analyze alerts and take necessary actions against theft.
10. End

METRICS USED IN THE PROJECT

Mean Squared Error (MSE) – Evaluates the Autoencoder by measuring the difference between predicted and actual power values (lower MSE is better).

Reconstruction Error – Used in the Autoencoder to detect anomalies; high error indicates potential energy theft.

Accuracy (CNN Model) – Measures how well the CNN predicts individual appliance power usage in Non-Intrusive Load Monitoring (NILM).

Precision & Recall (Theft Detection) – Precision ensures fewer false alarms, while recall ensures actual theft cases are detected.

RESULT & ANALYSIS

Energy Theft Detection – The Autoencoder model successfully identified anomalies in power consumption, flagging unusual spikes as potential theft cases.

Appliance-Level Monitoring – The CNN model effectively disaggregated total power usage, providing accurate insights into individual appliance consumption patterns.

Real-Time Monitoring & Alerts – The Flask-based dashboard displayed real-time power consumption, with alerts generated for suspicious activities.

```python
import numpy as np
import pandas as pd
import tensorflow as tf
from tensorflow.keras import layers, models
import matplotlib.pyplot as plt
import seaborn as sns
import random
import time
import tkinter as tk
from tkinter import ttk
from flask import Flask, render_template, jsonify

# Simulate smart grid power consumption dataset
num_houses = 5
num_appliances = 6
num_days = 30

def generate_smart_grid_data():
    appliances = ['Fridge', 'AC', 'TV', 'Lights', 'Washing Machine', 'Microwave']
    data = []
```

```python
for house in range(1, num_houses+1):
    for day in range(1, num_days+1):
        total_power = 0
        appliance_data = {}
        voltage = random.uniform(220, 240)
        current = random.uniform(5, 15)
        power_factor = random.uniform(0.8, 1.0)
        energy_theft = random.choice([0, random.uniform(2, 8)])  # Increased theft difference

        for app in appliances:
            expected_usage = random.uniform(1, 4) * (random.randint(5, 12))  # Slightly larger range
            actual_usage = expected_usage + random.uniform(-1, 1)  # Greater expected-actual variation
            appliance_data[f'Expected_{app}'] = expected_usage
            appliance_data[f'Actual_{app}'] = actual_usage
            total_power += actual_usage

        data.append({
            'House': house,
            'Day': day,
            'Voltage': voltage,
            'Current': current,
            'Power Factor': power_factor,
            'Total Power': total_power + energy_theft,
            'Energy Theft': energy_theft,
            **appliance_data
        })
    return pd.DataFrame(data)

# Generate and save dataset
dataset = generate_smart_grid_data()
dataset.to_csv("smart_grid_power_usage.csv", index=False)
print("Smart grid dataset generated and saved.")

# Load dataset
data = pd.read_csv("smart_grid_power_usage.csv")

# Autoencoder for anomaly detection
def build_autoencoder():
    model = models.Sequential([
        layers.Dense(16, activation='relu', input_shape=(4,)),
        layers.Dense(8, activation='relu'),
        layers.Dense(16, activation='relu'),
        layers.Dense(4, activation='sigmoid')
    ])
    model.compile(optimizer='adam', loss='mse')
    return model

autoencoder = build_autoencoder()
autoencoder.fit(data[['Total Power', 'Voltage', 'Current', 'Power Factor']],
                data[['Total Power', 'Voltage', 'Current', 'Power Factor']], epochs=50)
```

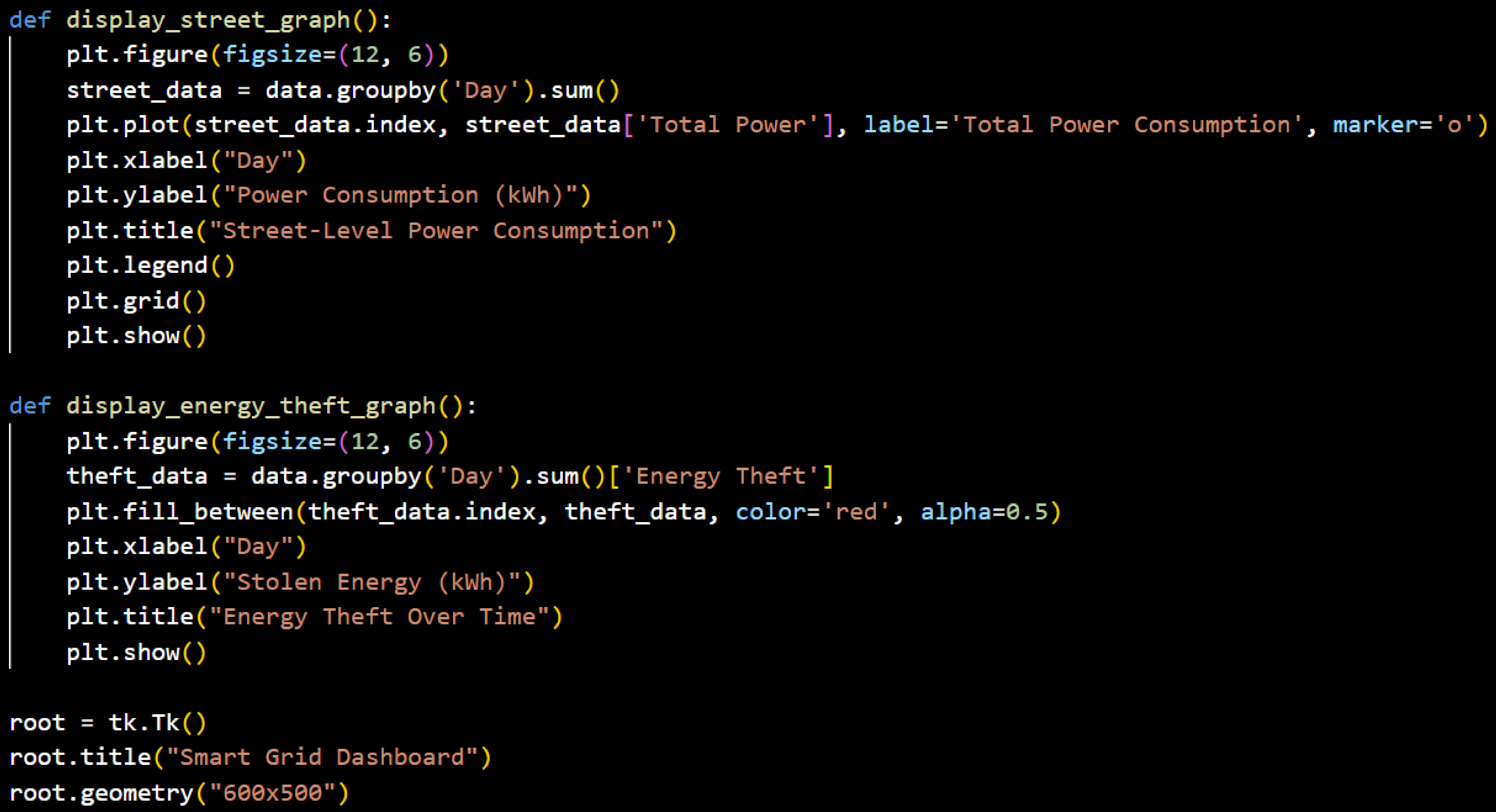
```python
# GUI for visualization
def display_house_graph(house):
    plt.figure(figsize=(12, 6))
    expected = [f'Expected_{app}' for app in ['Fridge', 'AC', 'TV', 'Lights', 'Washing Machine', 'Microwave']]
    actual = [f'Actual_{app}' for app in ['Fridge', 'AC', 'TV', 'Lights', 'Washing Machine', 'Microwave']]
    house_data = data[data['House'] == house].mean()

    x_labels = ['Fridge', 'AC', 'TV', 'Lights', 'Washing Machine', 'Microwave']
    expected_vals = [house_data[col] for col in expected]
    actual_vals = [house_data[col] for col in actual]

    x = np.arange(len(x_labels))
    width = 0.35

    fig, ax = plt.subplots()
    ax.bar(x - width/2, expected_vals, width, label='Expected')
    ax.bar(x + width/2, actual_vals, width, label='Actual')

    ax.set_xlabel("Appliance")
    ax.set_ylabel("Power Consumption (kWh)")
    ax.set_title(f"House {house} Power Consumption")
    ax.set_xticks(x)
    ax.set_xticklabels(x_labels)
    ax.legend()
    plt.show()
```



```python
def display_street_graph():
    plt.figure(figsize=(12, 6))
    street_data = data.groupby('Day').sum()
    plt.plot(street_data.index, street_data['Total Power'], label='Total Power Consumption', marker='o')
    plt.xlabel("Day")
    plt.ylabel("Power Consumption (kWh)")
    plt.title("Street-Level Power Consumption")
    plt.legend()
    plt.grid()
    plt.show()

def display_energy_theft_graph():
    plt.figure(figsize=(12, 6))
    theft_data = data.groupby('Day').sum()['Energy Theft']
    plt.fill_between(theft_data.index, theft_data, color='red', alpha=0.5)
    plt.xlabel("Day")
    plt.ylabel("Stolen Energy (kWh)")
    plt.title("Energy Theft Over Time")
    plt.show()

root = tk.Tk()
root.title("Smart Grid Dashboard")
root.geometry("600x500")
```

```python
tk.Label(root, text="Select Visualization:", font=("Arial", 12)).pack()
options = {"House Power Consumption": display_house_graph,
           "Street-Level Consumption": display_street_graph,
           "Energy Theft Trends": display_energy_theft_graph}

graph_selection = ttk.Combobox(root, values=list(options.keys()))
graph_selection.pack()

tk.Button(root, text="Display Graph", font=("Arial", 12), command=lambda: options[graph_selection.get()]() if graph_selection.get() else No

tk.Label(root, text="Select House for Power Graph:", font=("Arial", 12)).pack()
house_selection = ttk.Combobox(root, values=[1, 2, 3, 4, 5])
house_selection.pack()

tk.Button(root, text="Display House Graph", font=("Arial", 12), command=lambda: display_house_graph(int(house_selection.get())) if house_se

root.mainloop()
```

OUTPUT

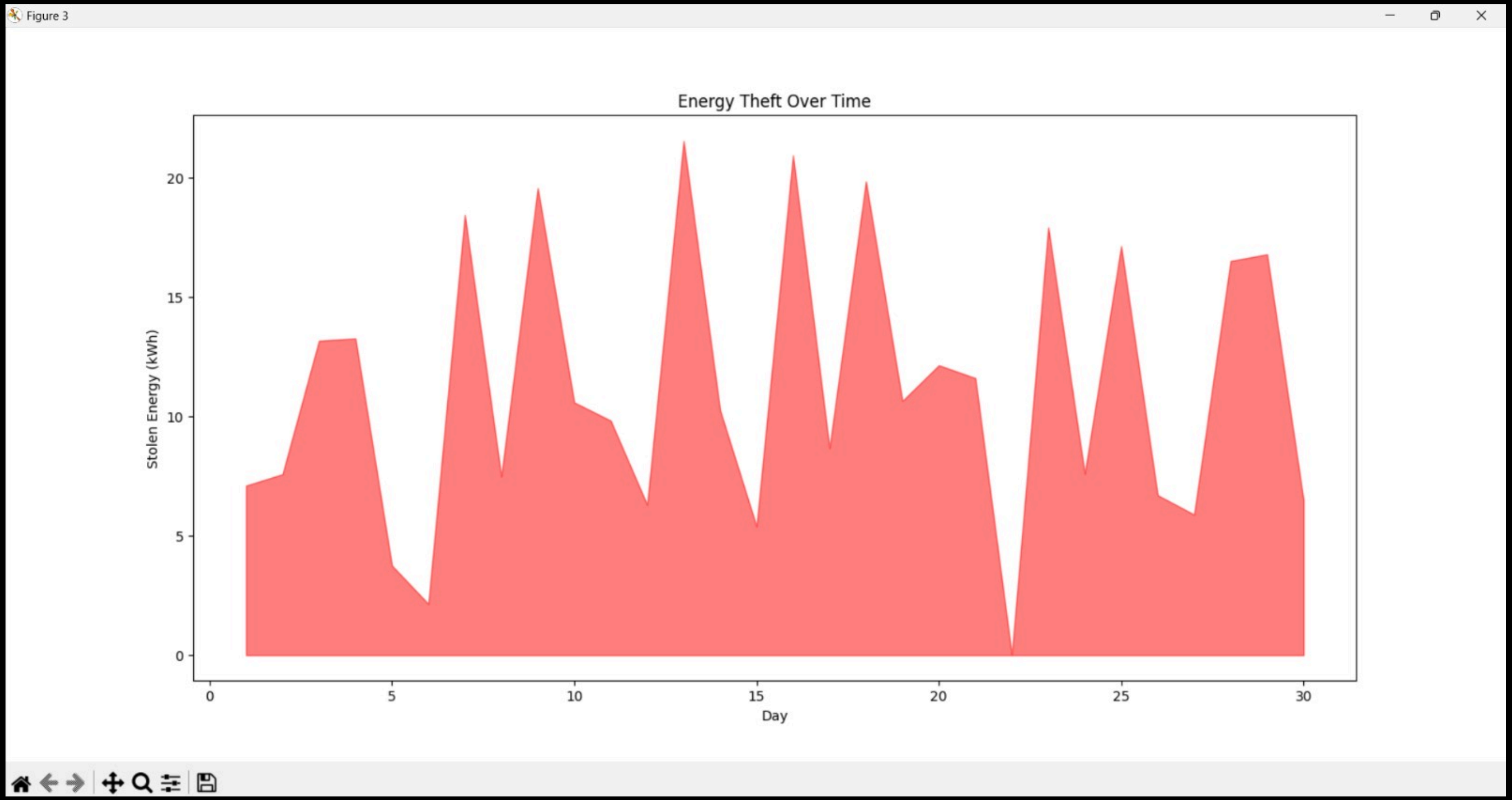


Figure 3
Energy Theft Over Time
Stolen Energy (kWh)
Day
0
5
10
15
20
25
30

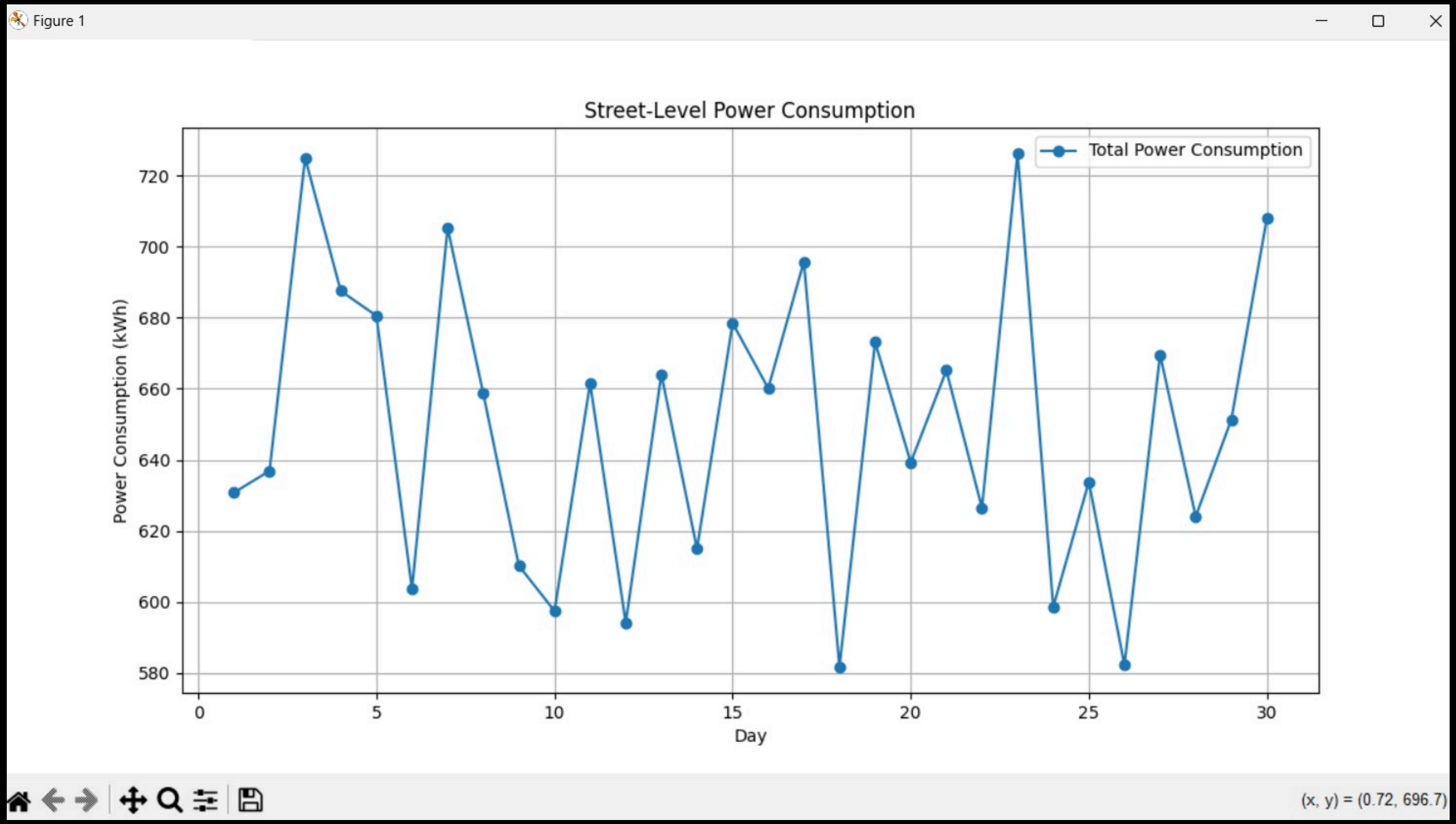


Figure 1
Street-Level Power Consumption
Total Power Consumption
Power Consumption (kWh)
Day
580
600
620
640
660
680
700
720
(x, y) = (0.72, 696.7)

CONCLUSION

This project leverages AI and Smart Grids to detect energy theft and monitor appliance-level power consumption using Autoencoders and CNNs. The Flask-based dashboard enables real-time monitoring and alerts, improving grid efficiency and security. Future enhancements include integrating IoT-enabled smart meters for more accurate detection

THANK YOU